Algorithmic Game Theory and Applications - Coursework 1

Finlay McAfee - s1220880

February 26, 2017

1

Let A be the bimatrix defining the 2-player strategic game G:

$$A = \begin{bmatrix} (6,8) & (2,9) & (3,8) & (2,8) \\ (0,5) & (2,3) & (2,6) & (8,4) \\ (7,0) & (2,7) & (4,4) & (4,3) \\ (2,3) & (5,3) & (2,5) & (5,4) \end{bmatrix} \tag{1}$$

1.1 Elimination

First we can eliminate the strongly dominated pure strategies from A to solve an equivalent but easier game.

For a pure strategy $x_i \in X_i$ to be strictly dominated there must exist another strategy x_i' such that, for all $x_{-i} \in X_{-i}$:

$$U_i(x_{-i}; x_i') > U_i(x_{-i}; x_i) \tag{2}$$

Note that this x_i' does not need to be a pure strategy. There are no pure strategies that strictly dominate any others in this game. But there are mixed strategies that dominate pure strategies.

Take x_2' to be $(0, \frac{1}{4}, \frac{3}{4}, 0)$. This mixed strategy strictly dominates $\pi_{2,1}$ and $\pi_{2,4}$ as:

$$U_2(\pi_{1,1}; x_2') = 8.25 \tag{3}$$
$$U_2(\pi_{1,2}; x_2') = 5.25 \tag{4}$$
$$U_2(\pi_{1,3}; x_2') = 4.75 \tag{5}$$
$$U_2(\pi_{1,4}; x_2') = 4.5 \tag{6}$$

Hence we can eliminate columns 1 and 4 from A:

$$A' = \begin{bmatrix} (2,9) & (3,8) \\ (2,3) & (2,6) \\ (2,7) & (4,4) \\ (5,3) & (2,5) \end{bmatrix} \tag{7}$$

By the same method we can choose x_1' to be $(0, 0, \frac{2}{3}, \frac{1}{3})$ which strictly dominates $\pi_{1,1}$ and $\pi_{1,2}$.

Therefore we can solve the following reduced matrix to find all Nash Equilibrium for the game G:

$$A'' = \begin{bmatrix} (2,7) & (4,4) \\ (5,3) & (2,5) \end{bmatrix} \tag{8}$$

1.2 Computing NE

It is clear by inspection that there are no pure Nash Equilibria. This is because, for every profile of pure strategies, there is always a player that is better off unilaterally switching to the other available pure strategy. By the definition of a Nash Equilibrium, this rules out pure NEs for the game G. By a similar argument there are no NE where one player plays a pure strategy and the other a mixed. However Nash's theorem tells us that a NE does exist, and as it must be a mixed NE, both players must play each strategy with positive probability. This means that each pure strategy, for both players, is a best response in a NE, by the Useful Corollary to Nash's Theorem.

Let $x_1^*(1)$, the probability of player 1 playing the strategy 1 in the NE, be a and $x_1^*(2)$ be $1 - a$. Let $x_2^*(1)$ be b and $x_2^*(2)$ be $1 - b$.

$$\begin{aligned} U_1^*(x_{-1}^*; \pi_{1,1}) &= U_1^*(x_{-1}^*; \pi_{1,2}) \quad (9)\\ 2b + 4(1-b) &= 5b + 2(1-b) \quad (10)\\ 4 - 2b &= 2 + 3b \quad (11)\\ b &= \frac{2}{5} \quad (12) \end{aligned}$$

Hence $x_2^* = (\frac{2}{5}, \frac{3}{5})$, or $(0, \frac{2}{5}, \frac{3}{5}, 0)$ under the original game.

$$\begin{aligned} U_2^*(x_{-1}^*; \pi_{2,1}) &= U_2^*(x_{-1}^*; \pi_{2,2}) \quad (13)\\ 7a + 3(1-a) &= 4a + 5(1-a) \quad (14)\\ 3 + 4a &= 5 - a \quad (15)\\ a &= \frac{2}{5} \quad (16) \end{aligned}$$

Hence $x_1^* = x_2^*$ (under the reduced game) and the profile of the NE for game G is

$$[(0, 0, \frac{2}{5}, \frac{3}{5}), (0, \frac{2}{5}, \frac{3}{5}, 0)] \tag{17}$$

As there was no loss of generality in these assumptions, it is clear that this is the only NE that exists for this game.

2

The LP corresponding to the given 2-player zero sum game can be specified by:

$$\begin{aligned}
\textbf{maximise} \quad & v \\
\textbf{subject to:} & \\
v - (7x_1 + 2x_2 + 6x_3 + 5x_4 + 2x_5) &\leq 0 \\
v - (x_1 + 6x_2 + 3x_3 + 5x_4 + 8x_5) &\leq 0 \\
v - (6x_1 + 8x_2 + 8x_3 + 4x_4 + 2x_5) &\leq 0 \\
v - (4x_1 + 3x_2 + 3x_3 + 7x_4 + 8x_5) &\leq 0 \\
v - (2x_1 + 5x_2 + 4x_3 + 4x_4 + 9x_5) &\leq 0 \\
x_1 + x_2 + x_3 + x_4 + x_5 &= 1 \\
x_i \geq 0 \quad \text{for} \quad j &= 1, ..., 5
\end{aligned}$$

This was computed using the `linprog` function in `MATLAB` in the following manner: Constraints 1,...,5 were encoded in the form $A\mathbf{x} \leq \mathbf{b}$ where:

$$\mathbf{b} = \mathbf{0} \tag{18}$$

$$A = \begin{bmatrix} 1 & -7 & -2 & -6 & -5 & -2 \\ 1 & -1 & -6 & -3 & -5 & -8 \\ 1 & -6 & -8 & -8 & -4 & -2 \\ 1 & -4 & -3 & -3 & -7 & -8 \\ 1 & -2 & -5 & -4 & -4 & -9 \end{bmatrix} \tag{19}$$

Constraint 6 was encoded in the form:

$$\text{beq} = 1 \tag{20}$$

$$Aeq = [0, 1, 1, 1, 1, 1] \tag{21}$$

And Contraint 7 was encoded as lower bounds for the variables $x_1, ..., x_5$
The program produced the following results:

$$v = 4.8333 \tag{22}$$

$$x_1 = 0 \tag{23}$$

$$x_2 = 0 \tag{24}$$

$$x_3 = 0.3333 \tag{25}$$

$$x_4 = 0.5 \tag{26}$$

$$x_5 = 0.1667 \tag{27}$$

Hence the minimax value of this game is 4.8333 and the optimal strategy for player 1 is $x_1^* = (0, 0, 0.3333, 0.5, 0.1667)$.

The variables of the dual of the linear program can be interpreted as the optimal strategy for player 2.

$$\begin{aligned}
\textbf{minimise} \quad & u \\
\textbf{subject to:} & \\
u - (7y_1 + 1y_2 + 6y_3 + 4y_4 + 2y_5) &\leq 0 \\
u - (2y_1 + 6y_2 + 8y_3 + 3y_4 + 5y_5) &\leq 0 \\
u - (6y_1 + 3y_2 + 8y_3 + 3y_4 + 4y_5) &\leq 0 \\
u - (5y_1 + 5y_2 + 4y_3 + 7y_4 + 4y_5) &\leq 0 \\
u - (2y_1 + 8y_2 + 2y_3 + 8y_4 + 9y_5) &\leq 0 \\
y_1 + y_2 + y_3 + y_4 + y_5 &= 1 \\
y_i \geq 0 \quad \text{for} \quad j &= 1, ..., 5
\end{aligned}$$

Computing this dual LP using `MATLAB` gives the following values:

$$u = 4.8333 \tag{28}$$

$$y_1 = 0.5556 \tag{29}$$

$$y_2 = 0.2778 \tag{30}$$

$$y_3 = 0 \tag{31}$$

$$y_4 = 0 \tag{32}$$

$$y_5 = 0.1667 \tag{33}$$

Hence player 2's optimal strategy is $x_2^* = (0.5556, 0.2778, 0, 0, 0.1667)$.

3

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{34}$$

(a)

The unique NE of the Matching Pennies game is given by $x^* = [(0.5, 0.5), (0.5, 0.5)]$

(b)

The experimental method was chosen for showing (but not proving) that the statistical mixed profile of this game approaches the above NE.

The tie breaking rule used was an arbitrary artifact of the implementation. Player 1 will chose tails in a tie and player 2 will chose heads.

Figures 1-4 show the estimated probability of the other player choosing the heads strategy under the statistical mixed strategy, for both players under all possible starting strategies.

Here follows the python implementation of the Matching Pennies Game:

```
#!/usr/bin/env python
import logging
import sys
import numpy as np
import matplotlib.pyplot as plt

HEADS = ’H’
TAILS = ’T’

logging.basicConfig(level=logging.WARNING)

class Player(object):
    ””” Class for a player in the matching pennies game
    ”””
    def __init__(self, matcher, new_game_policy):
        ”””
        Player 1 stores variables for estimating Player 2 and vice versa
        e.g. p1.N is the number of heads played by player 2
        ”””
        self.N = 0.0
        self.M = 0.0
        self.total = 0.0
        self.matcher = matcher
        self.new_game = True
        self.new_game_policy = new_game_policy

    def play(self):
        if self.new_game:
            self.new_game = False
            return self.new_game_policy
        if self.N > self.M:
            if self.matcher:
                return HEADS
            else:
                return TAILS
        else:
            if self.matcher:
                return TAILS
            else:
                return HEADS

    def update(self, other_players_move):
        if other_players_move == HEADS:
            self.N += 1
        else:
```

```python
            self.M += 1
        self.total += 1

    def get_sigma(self):
        return (self.N / self.total, self.M / self.total)

def matching_pennies_game(num_iterations, starting_policy):
    player_1 = Player(matcher=True, new_game_policy=starting_policy[0])
    player_2 = Player(matcher=False, new_game_policy=starting_policy[1])
    player_1_sigma_0 = []
    player_2_sigma_0 = []
    for i in xrange(num_iterations):
        # Each player moves
        player_1_move = player_1.play()
        player_2_move = player_2.play()
        # Each player updates their estimates
        player_1.update(player_2_move)
        player_2.update(player_1_move)
        if i % 10 == 0:
            player_1_sigma = player_1.get_sigma()
            player_1_sigma_0.append(player_1_sigma[0])

            player_2_sigma = player_2.get_sigma()
            player_2_sigma_0.append(player_2_sigma[0])

            logging.info("Player 1's estimate of player 2 sigma: {0}".format(player_1_sigma))
            logging.info("Player 2's estimate of player 1 sigma: {0}".format(player_2_sigma))

    x_axis = np.arange(0, num_iterations, 10)
    logging.debug(x_axis.shape)
    logging.debug(len(player_1_sigma_0))
    plt.figure(1)

    plt.title('Estimated Sigma')
    plt.xlabel('Iteration')
    plt.ylabel('Probability of Heads')
    plt.plot(x_axis, player_1_sigma_0, label='Player 1')
    plt.plot(x_axis, player_2_sigma_0, label='Player 2')

    plt.legend()
    plt.show()

if __name__ == "__main__":
    num_iterations = int(sys.argv[1])
    starting_policy = []

    if sys.argv[2] == 'heads':
        starting_policy.append(HEADS)
    else:
        starting_policy.append(TAILS)

    if sys.argv[3] == 'heads':
        starting_policy.append(HEADS)
    else:
        starting_policy.append(TAILS)

    matching_pennies_game(num_iterations, starting_policy)
```

Listing 1: question3.py

Figure 1: Player 1: Started with Heads, Player 2: Started with Heads

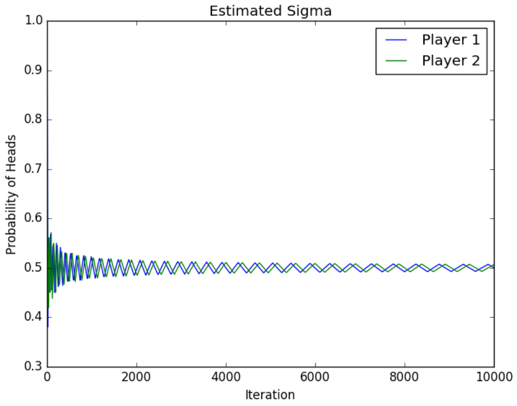


Figure 2: Player 1: Started with Heads, Player 2: Started with Tails

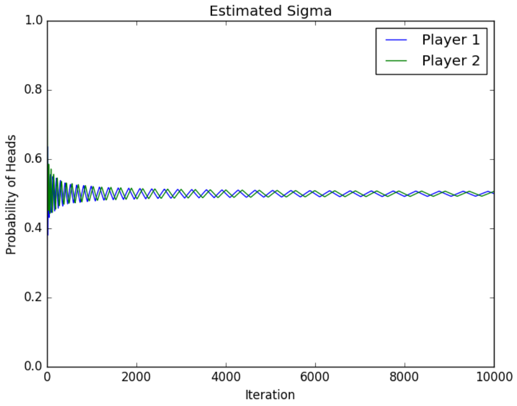


Figure 3: Player 1: Started with Tails, Player 2: Started with Heads

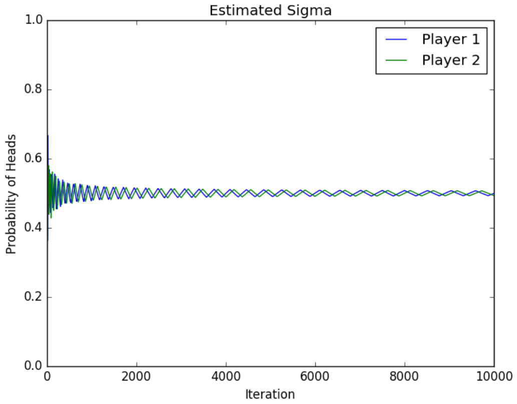
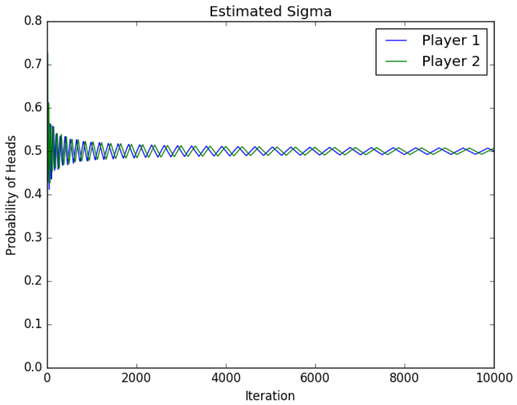


Figure 4: Player 1: Started with Tails, Player 2: Started with Tails



4

(a)

To show that there exists a strategy x_1 for player 1 that, if commited to, will increase their payoff to strictly greater than 6, first note that player 2 will respond with pure strategy 2 whenever $x_1(2) > x_1(1)$, as this will maximise their expected payoff. In the case of $x_1 = (0.25, 0.75)$ the expected payoff is player 2 chooses strategy 1 will be 2.25 and 2.75 for strategy 2. Now also note that, given player 2 is playing pure strategy 2, the expected payoff for player 1 will always be strictly greater than 6 by playing strategy 1 with non-zero probability, hence all mixed strategies

$$x_1 \quad s.t. \quad x_1(1) > 0, x_1(2) > x_1(1)$$

achieve a payoff > 6.

(b)

Based on the above reasoning and the new information about player 2's 'kind' strategy in the indifferent case, it is clear that the mixed strategy that maximises player 1's payoff is the one where he plays strategy 1 with the highest probability under the new conditions

$$x_1 \quad s.t. \quad x_1(1) > 0, x_1(2) \geq x_1(1)$$

, which is $x_1 = (0.5, 0.5)$.

(c)